Session 703

Project Builder In-Depth

Mike Ferris
Manager, Project Builder

Introduction

- Advanced features of Project Builder
  - Focus on the build system
- Simple projects are simple…
  - But what if your project isn't so simple?
- That's what this session is about

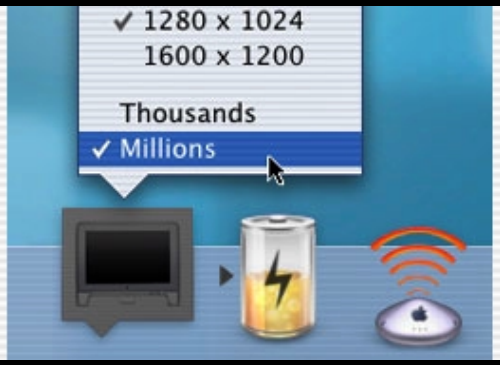
Why “In Depth”?

- Large projects can be complicated
- An IDE must scale to handle this complexity
  - Current needs often go beyond templates
  - Flexibility and extensibility anticipate the future
- Critical area is the Build System
  - Advanced features in other areas are important
- Bottom line: must be able to build your project

What You'll Learn

- Importing Sources into Project Builder
  - CodeWarrior projects
  - Other existing source code (Makefile based projects)
- Advanced Build System features
  - Targets—define products to build
  - Build Styles—variations on a target
  - Build Phases—target's basic building blocks
  - Legacy Targets—when all else fails

Importing Source



Importing Source

- Starting a new project
  - Use project templates for new projects
  - But what if you have existing code?
- Importing Sources into Project Builder
  - Import CodeWarrior projects
  - Import ProjectBuilderWO projects
  - Add source trees to a project directly

Importing From CodeWarrior

- Make sure to “Save project entries using relative paths”
- Export project as XML
- Use Import Project…in Project Builder
- Many things will be imported automatically
  - Group structure and files
  - Most of the target definitions
- Hand tuning of resulting project is often needed

Importing Source Trees

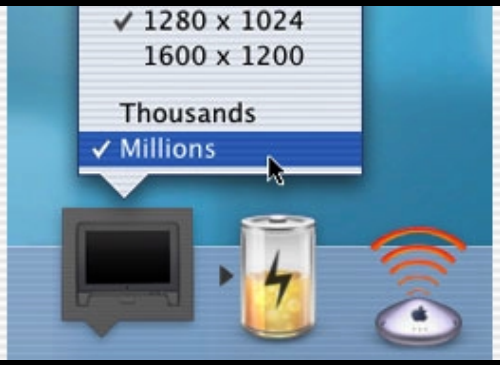
- Folders can be dragged in from Finder
- Steps for importing:
  - First create your target
  - Drag folder into Files and Groups pane
  - Finish configuring target

Demo

Importing Source

Anders Bertelrud
Senior Engineer, Project Builder

Build Styles



Anders Bertelrud
Senior Engineer, Project Builder

Build Styles

- Build targets in different ways
- Examples:
  - Debugging symbols on or off
  - Assertions enabled or disabled
  - Different optimization levels
  - Alternate algorithms
  - Features enabled or disabled

How Build Styles Affect the Build

- A build style changes settings at build time
- Each change can either:
  - Override the target's setting
    - **DEBUGGING_SYMBOLS = NO**
  - Append to the target's setting
    - **OTHER_CFLAGS += -DUSE_HASH_TABLES**
- Target settings can refer to build style settings
- Circular references are not permitted

Precedence of Build Settings

Using Build Styles

- The IDE always has one active build style
- The pbxbuild command-line tool allows greater flexibility:
  - pbxbuild install
  - pbxbuild install -buildstyle “Deployment”
  - pbxbuild -buildstyle “UsingHashTables”
  - pbxbuild -activebuildstyle

Default Build Styles

- New projects have two default build styles:
  - "Development" style for debug builds
  - "Deployment" style for final builds
    - pbxbuild install -buildstyle "Deployment"
- A project can have any number of build styles

Demo

Build Styles

Christian Molick
Build Environment Engineer

Build Phases

Christian Molick
Build Environment Engineer

Build Phases

- Phases define the steps of a build
- Phases can be added or deleted
- New files are added to the proper phase
- Files can be:
  - Added to or removed from phases
  - Moved within or between phases

Standard Build Phases

- Headers
  - Copied and cloned
- Bundle Resources
  - Copied, including property list files
- Sources
  - Derivation
  - Compilation

Standard Build Phases

- Java Archive Files
  - Constructs the hierarchy
- Frameworks and Libraries
  - Linking
  - Archiving Java hierarchy
- Resource Manager Resources
  - Rez and ResMerger

Custom Build Phases

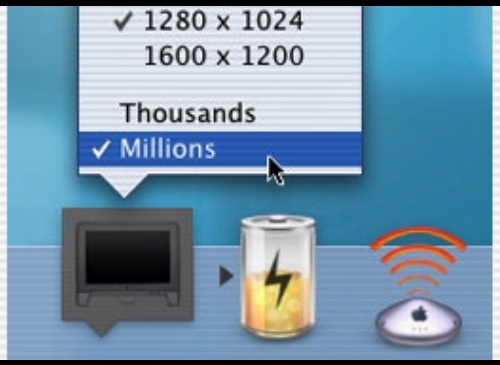
- Copy Files Phases
  - Special product contents
    - Read me files
  - Products including other products
    - Application plug-ins
    - Embedded frameworks
- Script Phases
  - Build process customization

Demo

Build Phases

Dave Ewing
Java Debugger Engineer

Legacy Targets



Dave Ewing
Java Debugger Engineer

Legacy Targets

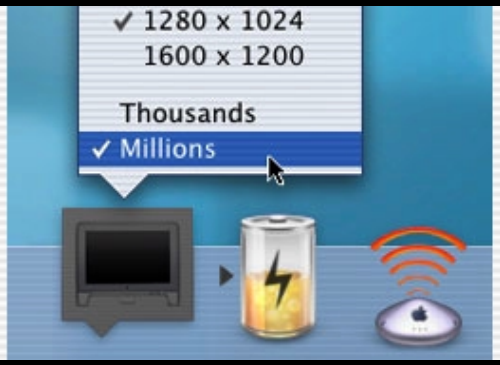
- Legacy targets invoke custom command line tools
- Legacy targets are used for…
  - Using a project's existing build system
    - Make, ant, other build systems
  - Generating source files needed by other targets
  - Pre- or post-processing on targets
  - Running custom scripts for other purposes

Examples

- Package disk images
- Build statistics logging
- Build notifications

Demo

Legacy Targets



Mike Ferris
Manager, Project Builder

Summary

- Importing existing code
  - From CodeWarrior/ProjectBuilderWO
  - Directly import source trees
- Flexible build system
  - Mostly automatic for common cases
  - Allows customization for complex projects
  - Build styles for building the same targets in different ways

For More Information

- Information pages for all Mac OS X tools
  http://developer.apple.com/tools/
- Project Builder web page
  http://developer.apple.com/tools/projectbuilder/

- Apple's mailing list master page
  http://lists.apple.com/

  Mailing lists:
  - **projectbuilder-users**
  - **cocoa-dev**
  - **carbon-dev**

Roadmap

503 Java Development Tools: Tools for the Java developer	Room A1 **Wed., 10:30am**
704 Compiler Technologies: Latest GCC enhancements	Room A2 **Thurs., 10:30am**
705 Apple Performance Tools: Fix leaks and accelerate	Room A2 **Thurs., 5:00pm**
706 Debugging on Mac OS X: Find and squash bugs	Hall 2 **Fri., 9:00am**
708 C++ Frameworks for Mac OS X: Building and using them	Room C **Fri., 2:00pm**
FF119 Feedback Forum: **Apple Developer Tools:** Give your feedback to the tools team!	Room J1 **Fri., 3:30pm**

Who to Contact

Godfrey DiGiorgi
Technology Manager for Development Tools
Apple Worldwide Developer Relations
ramarren@apple.com

macosx-tools-feedback@group.apple.com
Mailing address for feedback

Q&A

Godfrey DiGiorgi
Technology Manager for Development Tools
ramarren@apple.com

http://developer.apple.com/events/urls.html

WWDC

Worldwide Developers Conference 2001